VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Poziční a paralelní kopírování dat
Positional and Parallel Data Copy

2012

Lukáš Chlebek

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Lukáš Chlebek**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Poziční a paralelní kopírování dat
Positional and Parallel Data Copy

Zásady pro vypracování:

Běžné aplikace pro kopírování dat provádí sekvenční kopírování souborů podle abecedního uspořádání. Vyššího výkonu u pevných disků by se dosáhlo minimalizací pohybů čtecích hlav. U pamětí typu Flash by mohl být výhodnější paralelní přístup. Úkolem je tedy vypracovat aplikaci pro systém Windows, která zahrne tyto varianty:

1. Sériové čtení se sériovým zápisem.
2. Sériové čtení s paralelním zápisem.
3. Paralelní čtení se sériovým zápisem.
4. Paralelní čtení s paralelním zápisem.

Student provede srovnání s konkurenčními aplikacemi pro všech devět kombinací čtení a zápisu na HDD, DVD a Flash médiích a navrhne vhodné velikosti vyrovnávací paměti.

Seznam doporučené odborné literatury:

Podle pokynů vedoucího bakalářské práce.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Pavel Krömer, Ph.D.**

Datum zadání: 19.11.2010

Datum odevzdání: 04.05.2012

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Dne: 17. 8. 2012 .....

„Souhlasím se zveřejněním této bakalářské/diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v bakalářských/magisterských programech VŠB-TU Ostrava."

Dne: 17. 8. 2012 .....

Podpis zástupce

Rád bych poděkoval všem, kteří mi pomohli. Především mému vedoucímu Ing. Pavlovi Krömerovi, Ph.D. za odbornou pomoc a konzultaci při vytváření této diplomové práce.

Abstrakt

Cílem této práce je vytvoření aplikace, která by umožňovala rychlejší kopírování souborů, než Průzkumník Windows. Teoretická část této práce se věnuje pevným diskům a flash pamětím, multitaskingem a vláknům, jinými programy zabývající se stejnou problematikou a jazyku C#. V praktické části je popsána aplikace, prováděné testy a jejich výsledky.

Klíčová slova: kopírování, soubory, C#, Windows, vlákna

Abstract

The aim of this work is to create an application that would allow faster copying files than Windows Explorer. The theoretical part of this work is dedicated to hard drives and flash memories, multitasking,threads and other programs with the similar functionality and the language C#. The practical part describes the application, performed tests and their results.

Keywords: copy, files, C#, Windows, threads

Seznam použitých zkratek

PC	- personal computer
RAM	- random-access memory
SSD	- solid-state drive
HDD	- hard disk drive
CRC	- cyclic redundancy check
CD	- compact disc
DVD	- digital video disc

Obsah

Seznam obrázků

1 Úvod

V dnešní době je samozřejmostí v každé domácnosti alespoň jeden počítač a téměř všechny používají operační systém Windows. Existují i jiné operační systémy, ale nejsou tak rozšířené jako produkty společnosti Microsoft. Uživatelé používají mnoho úkonů při práci s PC. Jedním z nejčastějších úkonů je kopírování souborů z jednoho místa disku na jiné místo, popř. kopírování mezi dvěma disky či z externího média na disk nebo naopak. Většina správců souborů používá stejnou techniku kopírování jako výchozí správce souborů ve Windows, Průzkumník. Jedná se o jednoduché sekvenční kopírování. V případě, že často kopírujete větší soubory a potřebujete tento úkon zrychlit, Průzkumník není dostatečně efektivní.

V této bakalářské práci se podíváme, jakým způsobem se soubory kopírují ve Windows a jakým způsobem by se toto kopírování dalo zrychlit. Samotným cílem této bakalářské práce je vytvořit aplikaci, která by dokázala kopírovat soubory rychleji než klasický Průzkumník Windows. Otestovat tuto aplikaci a porovnat s jinými používanými programy, které se zabývají touto tématikou. Aplikace by měla umožňovat především rychlejší kopírování.

2 Cíl práce

Tato kapitola se bude zabývat úvodem do problematiky kopírování souborů a návrhy jak zrychlit kopírování souborů pomocí specifického softwaru.

2.1 Disky

K uchování informací již od počátku počítačů sloužily speciální média, která se později začala nazývat disky. První krůčky ovšem směřovaly od děrných štítků, které později zpracovávaly tehdejší počítače. Běžná kapacita děrných štítků byla kolem 80 znaků, což není v porovnání s dnešními disky moc. Později se začaly používat magnetické pásky, které měly mnohem větší kapacitu, než děrné štítky. Průměrná kapacita byla 128 znaků na palec délky. Problém byl v nalezení určité části dat. Pokud jsme chtěli načíst pouze určitou část dat, museli jsme projet celou pásku až k danému místu, což znamenalo velkou časovou náročnost pro vyhledávání konkrétní informace.

S postupem času se začaly objevovat disky dnešního typu, které sice neměly ani zdaleka takovou kapacitu, ale pracovaly na stejném principu. V dnešní době se používají různé variace záznamových médií, používáme pevné disky, pružné disky, optické disky, flash disky, SSD disky a další.

Pružné disky nebo-li floppy disky jsou magnetická média, které byly hojně rozšířeny v druhé polovině 20. století. Jednalo se o diskety, které byly populární právě díky levné pořizovací a výrobní ceně samotných disků, ale i čteček. Diskety sloužily pro přenos menšího objemu dat, ale přesná velikost floppy disků se lišila podle samotné velikosti diskety. Složení diskety je jednoduché. Plastový obal chrání vnitřní nosič s magnetickou vrstvou. Typická disketa se skládá z přepínače ochrany proti zápisu, upínací části, kde je vyznačen první sektor, krytky čtecího otvoru, plastického krytu, papírové vložky, magnetického nosiče a sektorů na disku.

Pevné disky, nebo-li anglicky Hard Disk Drive, představují nyní hlavní část médií, na kterých jsou uložena data. V téměř každém osobním počítači jsou pevné disky. Využívají magnetickou indukci pro dočasné či trvalé zaznamenání a reprodukování dat.

Disk se skládá z diskových ploten a hlav. Disk obsahuje keramické či kovové desky zvané plotny. Typický disk dnes již obsahuje více ploten. Obvodové rychlosti ploten se pohybují dokonce okolo 100 km/h. Výkon disku je určen otáčkami disku, hustotou záznamu na jednotku plochy a rychlosti vystavovacího mechanismu.

Hlavy jsou zařízením, které zajišťuje samotné čtení, a zápis dat. Na každou použitou plotnu se používají právě dvě hlavy, protože na plotně jsou data zapsána oboustranně. Vystavovací mechanismus nastavuje hlavy na danou pozici, kde jsou uložena data, ke kterým chceme přistupovat. Vystavovací mechanizmus je tvořen elektromagnetem. Při čtení či zápisu dat musíme nejprve vystavit čtecí hlavu na požadované místo, poté počkat, až se hlavy stabilizují kvůli setrvačnosti a nakonec počkat na pootočení disku na dané místo. Nyní je disk schopen číst či zapisovat data na všech plotnách bez pohybu čtecích hlav. V současné době je přístupová doba u disků okolo 8.5m/s při rychlosti 7200 otáček za minutu.

Největší výhodou pevných disků je pořizovací cena vzhledem k objemu dat, které jsou možné zaznamenat a přijatelné rychlosti čtení a zápisu. Ale na druhé straně provozovací cena pevných disků není tak příznivá, protože mají vysokou spotřebu elektrické energie a také jsou náchylné na mechanické poškození.

Flash paměť je dalším způsobem uložení dat. Flash paměti jsou nevolatilní programovatelné paměti, které mají libovolný přístup (random memory access). Paměť je vnitřně organizovaná po blocích, které jsou naprogramovatelné. Tyto paměti jsou často využívány k uložení firmwaru.

Flash paměti se skládají z unipolárních tranzistorů s plovoucími hradly. Existují dva typy hradel a to hradla, které propouští elektrony dále a hradla, které dokáží elektrony zadržet, protože jsou izolované a tím se informace uloží.

V dnešní době se hojně využívají flash paměti v kapesních přehrávačích MP3, přenosných USB discích, ale také jsou využívány v SSD discích jako vestavěné paměti. SSD diskům se budou věnovat následující odstavce.

SSD disky nebo-li solid state drives jsou nejčastěji založeny na nevolatilních pamětech NAND flash, alé také můžeme najít i SSD disky, které jsou založeny na DRAM, což znamená, že mají zálohované napětí. Existují také kombinace výše zmíněných. SSD disky nepoužívají pohyblivé mechanické části. V budoucnosti se počítá s postupným nahrazením pevných disků právě těmito disky. SSD disky využívají rozhraní SATA nebo sběrnici PCI-E.

Oproti pevným diskům mají SSD disky mnoho výhod, mezi které patří rychlý náběh, náhodný přístup k datům, čímž se snižuje přístupová doba, větší přenosové rychlosti, tichý provoz a nižší spotřeba. Velikou výhodou oproti pevným diskům je také mechanická a magnetická odolnost.

Ovšem existují také nevýhody, mezi které řadíme vyšší cenu za jednotku prostoru, omezený počet přepisů, ale pro běžného uživatele to nemá příliš velký význam. Existují také možné komplikace se zabezpečením, která se projevovala například zápisem na několik míst při přepisu.

NAND flashe na SSD discích jsou organizovány do stránek, které se později organizují do bloků. Pro přepis je nutno načíst celý blok do vyrovnávací paměti, kde se později změní, na disku se vymaže a poté se zpětně zapíší. Problém je menší, když zapisujeme sekvenčně než při náhodném zápisu.

Problém SSD disků spočívá v přepisech a proto existuje několik možných řešení. Oficiální kapacita SSD disku může být menší, než je jeho skutečná kapacita. Příkaz TRIM, který byl implementován do operačních systémů, jako jsou distribuce UNIXu a Windows 7. Tento příkaz umožňuje souborovému systému sdělit, které bloky nejsou používány a tak je lze považovat za prázdné. Další možností je, že novější řadiče SSD disků uvolňují stránky na základě dostupných dat. Ještě si popíšeme jednu možnost, která k minimalizaci počtu přepisu stránek využívá možnost zapsání na jinou pozici a přesouvá dlouho neměněné stránky.

2.2 Úvod do problematiky

Podíváme se, jak by se dala zrychlit práce s uložištěm dat v PC obecně. Dále se budeme věnovat magnetickým diskům neboli HDD.

Nejefektivnější práce s magnetickým diskem by znamenala maximálně omezit pohyby čtecích hlav disku, které prodlužují práci na požadovaných úkonech. To by ale znamenalo, že musíme znát přesnou polohu dat na disku a podle této polohy s nimi pracovat. Například při načítání souboru by se nečetl soubor sekvenčně od začátku do konce, ale četl by se podle fyzického rozložení na disku. Od středu diskové plotny k jejímu okraji či naopak. Naopak při ukládání by se soubor mohl ukládat na více ploten do stejné stopy najednou, tudíž by se čtecí hlavy nemusely pohybovat tak často. Toto by mohlo velmi omezit pohyby čtecích hlav, a proto i značně zrychlit práci s diskem. Tato myšlenka by se mohla aplikovat i na práci s více soubory. Při čtení více souboru by se načítaly podle fyzického rozložení na disku od středu diskové plotny k jejímu okraji či naopak. Při zápisu by se postupovalo obdobně jako u práce s jedním souborem. Takovýto způsob práce s diskem by měl být velice efektivní, ale na druhou stranu velice náročný na správný provoz. Při zápisu by se muselo vyhledat místo na disku, kam se má vůbec zapisovat, které by vyhovovalo požadavkům na efektivitu, tzn. dostatek místa bez již využitých alokačních bloků. Při čtení by se muselo zjistit, kde jsou tato data fyzicky uložena na disku, aby se mohla efektivně načítat. Další problém by mohl nastat při větším zaplnění disku. Mohlo by dojít volné místo, dostatečně veliké pro spojité naalokování, tudíž by se sáhlo pro alokační bloky, které by již nebyly na disku uloženy za sebou, a mohlo by docházet k externí fragmentaci, která práci s diskem velmi zpomaluje. Takováto metoda práce s diskem by měla větší množství problémů, protože k její funkci je zapotřebí znát fyzické rozložení souborů na disku a mít možnost ukládat na konkrétní fyzické místo. Tuto možnost nám dnešní hardware vůbec neumožňuje. A proto zrychlit práci s diskem vůbec nemáme možnost. Z tohoto důvodu se musíme uchýlit k softwarovému řešení zrychlení kopírování s magnetickým diskem.

Jedna ze softwarových možností zrychlení by mohla záležet na způsobu kopírování podle použitých pamětí a médií, ze kterých a na které by se kopírovalo. Při kopírování na jednom pevném disku je nejrychlejší způsob, jak provést kopírování následující: načíst soubor nebo jeho část do operační paměti a následně uložit. Takto funguje běžné sekvenční kopírování, které se používá v systémech Windows. Nicméně i tento triviální úkon by se dal zrychlit. Kdyby se načítaly větší bloky, nemusely by se čtecí hlavy tolik pohybovat. To by mělo za následek zrychlení celého kopírování. V dnešní době není problém do paměti načítat více než sto megabajtů. V případě kopírování mezi dvěma disky lze dosáhnout zrychlení tak, že se bude zároveň číst z jednoho disku do paměti a ve stejném okamžiku zapisovat z již načtené paměti na druhý disk. Teoreticky by tímto způsobem měla stoupnout rychlost kopírování téměř dvojnásobně, což je značné zrychlení. V případě kopírování z optického média se bude chovat stejně jako kopírování mezi dvěma pevnými disky. Při použití elektronických pamětí jako flash disk nebo SSD disk lze použít paralelní přístupy, které jsou u mechanických disků velice nepraktické a zpomalují. Toto by mělo význačně přispět k zvýšení efektivity využití těchto pamětí. Je třeba určit, co by se mělo paralelizovat a do jaké míry. Zda-li se má kopírování jednoho souboru provádět paralelně a kopírovat jeden po druhém, nebo každý soubor kopírovat v jednom vlákně a při kopírování více souborů by se četlo/zapisovalo více souborů paralelně. Nebo obě možnosti zároveň, což znamená kopírovat více souborů najednou a každý navíc ve vláknech.

Při kombinaci magnetického/optického disku a disku s flash pamětí by bylo nejefektivnější z magnetického/optického disku číst jedním vláknem data sériově a na disk s flash pamětí zapisovat paralelně, v případě zápisu na magnetický disk zapisovat jedním vláknem data sériově a z disku z flash pamětí číst paralelně.

Při práci s vlákny je také třeba zjistit, jaké nastavení je nejefektivnější, jaká je nejlepší velikost vyrovnávací paměti, jaký je vhodný počet vláken a ideální počet vyrovnávajících pamětí. Malá velikost vyrovnávací paměti může zapříčinit váznutí vláken z důvodu nedostatku dat, naopak příliš velká paměť může způsobit zahlcení operační paměti. Může dojít i k situaci, kdy je vyrovnávací paměť větší než operační. Tudíž se kopírování může zpomalit, popř. může nastat i chyba. Když bude ve vyrovnávací paměti větší počet dat, může dojít ke zpomalení procesu kopírování mezi dvěma médii, protože by se mohlo pouze číst a pak až zapisovat. Ovšem rychlejší je způsob, kdy se může zároveň číst i zapisovat. Pravděpodobně by bylo nejvýhodnější zvolit velikost vyrovnávací paměti podle velikosti zpracovávaného souboru a poctu vláken, z důvodu aby operaci načíst/uložit mohlo zpracovávat víc vláken zároveň. Počet vláken je docela nejistá a teoretická věc, závisela by na testech. Počet vyrovnávacích pamětí by měl být větší než dvě na jedno použité vlákno, aby nedocházelo k zastavení některého vlákna jak z důvodu plného bufferu, tak i z důvodu prázdného bufferu.

2.3 Multitasking

Dnes se využívají téměř výhradně počítače s víceúlohovými operačními systémy. Mezi tyto systémy patří Microsoft Windows, Linux i Mac OS. Mezi jednoúlohové operační systémy patří například MS DOS. V této kapitole se podíváme, jakým způsobem se zpracovává více operací najednou.

Víceúkolové systémy jsou takové systémy, které dokáží vykonávat více úkonů zdánlivě najednou. Díky tomu můžeme na počítači například poslouchat hudbu, prohlížet internet, stahovat data, kopírovat soubory a to vše najednou. Opravdového paralelního zpracovávání by se dalo dosáhnout tak, že by každý proces zpracovával jeden procesor. Tímto způsobem by byl omezen počet aktuálně běžících programů (procesů). V případě, že by se některý procesor nevyužíval, nedokázal by pomoct ostatním procesorům a urychlit vykonávání. Navíc by takovéto řešení bylo dost nákladné a složité, také dost možná i rozměrné, tudíž nevyhovující. Z tohoto důvodu se používá multitasking neboli pseudoparalelizmus. Multitasking znamená, že procesor dokáže provádět více operací zdánlivě najednou. Procesor podporující multitasking pracuje tak, že každý běžící proces se zpracovává po částech a ne celý najednou. Díky tomu se tyto procesy mohou střídat rychle za sebou. Tyto procesy (kontext) se střídají typicky 100x až 1000x za sekundu. Procesor dokáže pracovat rychleji než ostatní zařízení v počítači, například operační paměť RAM nebo HDD. Proto má procesor vnitřní vyrovnávací paměť cache, která je sice značně menší, ale o to rychlejší. Tato paměť slouží k předběžnému načítání dat, aby procesor nemusel čekat do doby, než se načtou data z RAM nebo z pevného disku. Navíc se doba vykonávání jednotlivých procesů mění, aby se maximálně využil výkon procesoru. Například se mění podle priority procesu, nebo jestli některý proces čeká na načtení dat z disku nebo z paměti RAM, tak se nečeká, než se data načtou, ale začne se vykonávat další proces. Díky multitaskingu pracují i vlákna, která se zpracovávají podobně jako jednotlivé procesy. Podrobněji si vlákna popíšeme v následující kapitole.

2.4 Vlákna

Nyní se podíváme na vlákna. Popíšeme si, k čemu slouží a jak pracují. V dnešní době, kdy už se nezvyšuje frekvence procesoru, ale zvyšuje se počet jader, má práce s vlákny větší význam.

Vlákno neboli thread je v informatice pojem používaný v souvislosti s odlehčeným procesem. V rámci jednoho procesu může běžet více vláken. Vlákna se používají pro snížení režie operačního systému, když přepínáme kontext. Sdílí kód, data a další zdroje, ale mají vlastní obsah registrů a zásobník

Přepínání kontextu slouží pro urychlení výpočtů procesoru a také zdánlivé dokončení operací současně. Než se budeme zabývat samotným přepnutím kontextu, musíme si vysvětlit, co znamená PCB. PCB (process control block) je struktura, která se zabývá procesem. PCB typicky zahrnuje identifikátory spojené s procesem, stav plánování procesu, obsahy registrů, plánovací informace (priorita, ukazatele na plánovací fronty, atd.), informace spojené se správou paměti (tabulky stránek), informace spojené s účtováním (spotřeba procesoru) a využití I/O zdrojů. PCB může být rozdělen do několika dílčích struktur. Při přepnutí kontextu dispečer odebere procesor procesu A a přidělí ho procesu B, což zahrnuje úschovu stavu některých registrů v rámci procesu A do PCB. Poté nastanou úpravy řídících struktur v jádře. Po dokončení tohoto úkonu nastane obnova řídících struktur procesu B z daného PCB. Posledním krokem je předání řízení procesu B na adresu, kde byl proces přerušen. Nyní běží na daném procesoru proces B, procesy se střídají, když plánovač (scheduler) předá řízení dispečerovi (dispatcher). Plánovač používá různé algoritmy k určení procesu, kterému se přidělí procesor. Obvykle to záleží na spotřebovaném procesorovém čase a době, která je nezbytná pro procesor, aby dokončil daný proces.

Vlákna můžeme rozdělit do tří skupin. Rozdělení probíhá na základě spravování operačním systémem. Vlákna na uživatelské úrovni představují první skupinu, druhá skupina jsou vlákna, která běží na úrovni jádra a poslední skupinu představuje kombinace výše zmíněných skupin.

Do první skupiny jsou řazeny vlákna na uživatelské úrovni. Jejich typickým prvkem je správa vláken vláknovou knihovnou. Vláknová knihovna spravuje vlákna na úrovni aplikačního procesu. Nepřepíná se kontext procesu, ani režim procesoru. Přepínání vláken a vlastní plánování je vždy specifické pro danou aplikaci. Výhodou této skupiny je nezávislost na podpoře vláken v operačním systému, ale také úplná kontrola procesu nad daným vláknem. Na druhou stranu ale existují i nevýhody v podobě zablokování všech vláken při volání služby jedním vláknem. Dále také neexistuje podpora souběžně běžících vláken v jednom procesu i za situace, kdy počítač disponuje více procesory.

Druhá skupina se zabývá vlákny, která běží na úrovni jádra. Jádro se tedy stará o celou údržbu a správu vláken v procesech a programátor na to nemá vliv. Samotné jádro se tedy stará o vytváření, rušení a plánování vláken. Odpadá problém při zablokování ostatních vláken při volání služby jedním vláknem, jak je tomu u vláken na uživatelské úrovni. Další nespornou výhodou je možnost využití více procesorů jedním procesem, ale správa vláken je mnohem náročnější než u vláken na uživatelské úrovni.

Poslední skupinu zaujímá kombinace vláken na uživatelské úrovni a vláknech na úrovni jádra. Tento způsob pojetí vláken je podporován na několika operačních systémech. Programátor má

možnost nastavit si, zda-li chce vlákna spravovat on sám z hlediska nastavení počtu vláken nebo to nechá nastavit systém automaticky.

Vlákna usnadňují vzájemnou komunikaci procesů, ale vznikají díky nim i problémy v souběhu, například race condition.

2.5 Shrnutí cíle práce

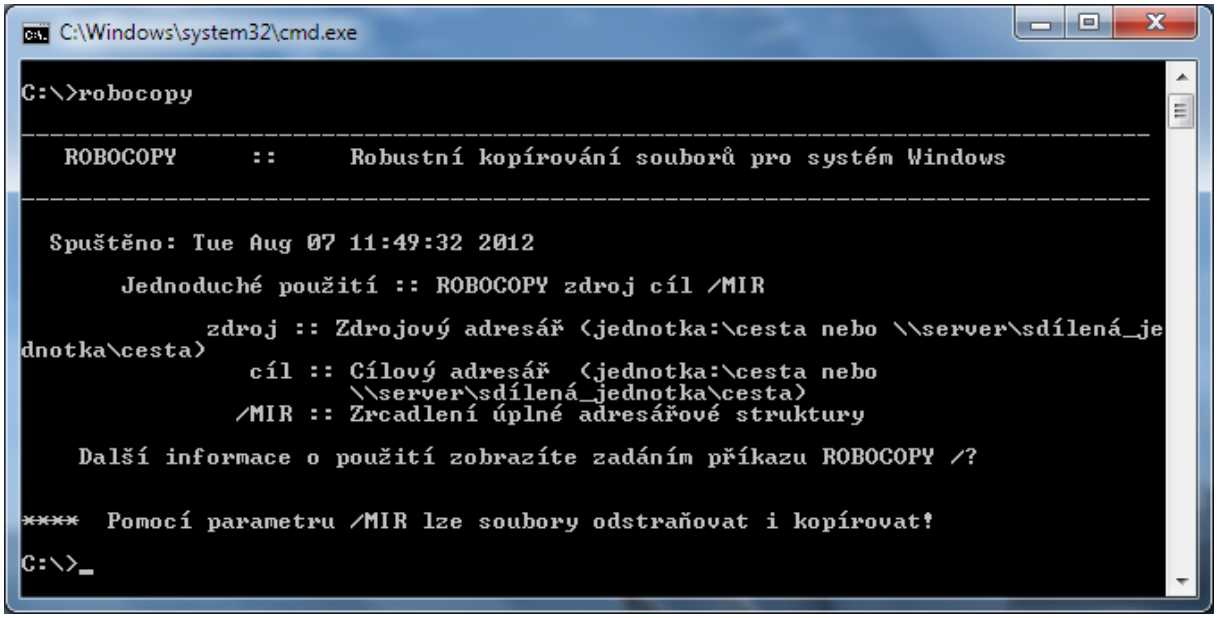
V předcházejících kapitolách jsme se seznámili s problematikou a s vlákny. Vlákna jsme si popisovali z důvodu, že se je pokusíme využít k řešení této práce. Chceme tedy využít paralelizmu za pomocí vláken pro zrychlení kopírování dat na disku, nebo mezi dvěma disky.

3 Programy k porovnávaní

Dnes již existují programy, které nabízí rozšířené možnosti kopírování nebo jeho zrychlení. Dále si představíme dva z nich. První je již standardní součástí Windows Vista a novějších a druhý, který zastupuje freewarové programy.

3.1 Robocopy

Systém Windows obsahuje nástroj s názvem Robocopy, který už je standardní součástí systému Windows Vista, Windows 7, Windows Server 2008 a Windows Server 2008 R2. Do starších verzí systémů Windows XP nebo Windows Server 2003 apod., lze doinstalovat pomocí Windows Server 2003 Resource Kit Tools. Jedná se o konzolovou aplikaci, která rozšiřuje příkazy COPY a XCOPY. Je možné si stáhnou grafické rozhraní pro jeho snadnější ovládání. Tento program má velké množství funkcí. Dokáže vybírat soubory nejen podle názvu nebo umístění v adresářové struktuře, která je určena ke kopírování, ale také umožňuje soubory vybírat podle atributů (například podle atributu archivovat a to nám usnadňuje vytvoření zálohy), kopírovat pouze soubory, které v cílovém adresáři chybí, nebo zkopírovat a přepsat jen ty, které se v adresáři nacházejí, atd. Lze nastavit, jestli program má kopírovat podadresáře či nikoliv, nebo také jestli se mají kopírovat časová razítka. V případě, že dojde k chybě, se dá nastavit počet opakování a doba čekání mezi opakováními. Zápis výsledků do souboru je zde taky k dispozici, spolu s bohatým nastavením formátu toho, co se má vůbec uložit do souboru, nebo jestli se má v případě již existujícího logovacího souboru výsledky připsat nebo soubor nahradit. Program umí i zrcadlit libovolný adresář, nebo celý disk, či synchronizovat dva adresáře. Pro práci se systémovými, či jinými soubory, které jsou neustále využívány, umí program kopírovat po restartu.

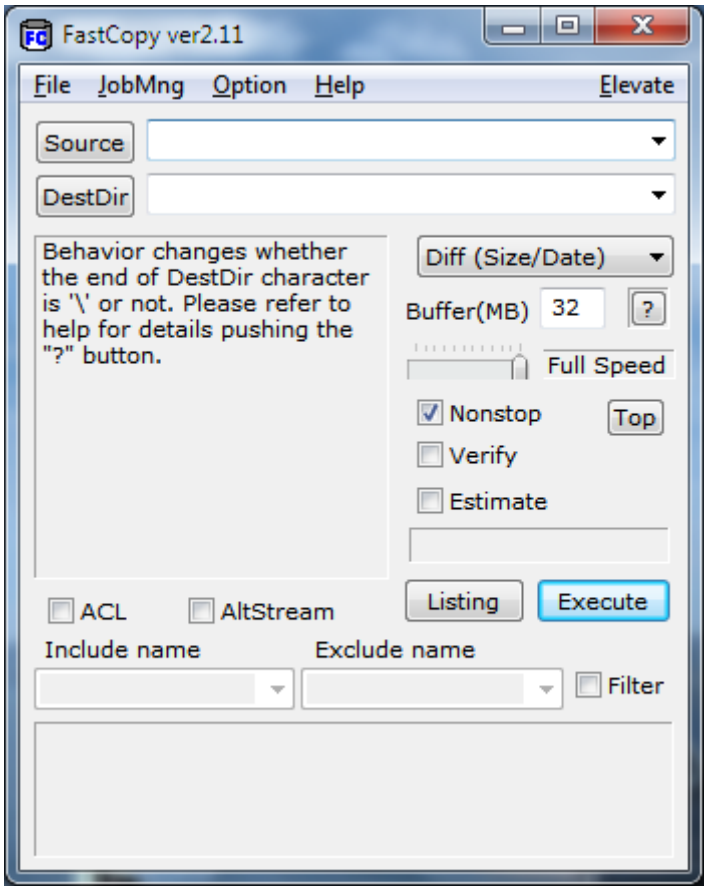


Obrázek 3.1: Nápověda příkazu robocopy

Program lze dobře využít pro zálohování dat díky tomu, že lze vytvořit job a umí kopírovat pouze soubory, které jsou označeny atributem archivovat. Mezi možnosti nastavení tohoto programu patří možnost kopírovat ve vláknech. Tato možnost nás bude zajímat.

3.2 FastCopy

Mezi freewarové programy patří například FastCopy. Umožňuje rychlé kopírování souborů a složek. Dokáže rozpoznat, jakou metodu kopírování použít, aby dokázal co nejrychleji zkopírovat zadané soubory. Díky tomu je ve všech případech rychlejší než průzkumník Windows. Rozpozná totiž zařízení, mezi kterými se bude provádět kopírování, tedy jestli se kopíruje v rámci jednoho disku, mezi diskem a optickým médiem (CD, DVD, Blu-Ray), nebo flash diskem, apod. Díky tomu dokáže vybrat nejvhodnější metodu ke kopírování. Při kopírování v rámci jednoho disku nejdříve načte část souboru do vyrovnávací paměti a poté až zapíše. Podobně funguje i běžné sekvenční kopírování, jako má například průzkumník Windows, ale tento program má značně větší vyrovnávací paměť, která se dá libovolně nastavit. Výchozí hodnota vyrovnávací paměti je 32 MB, což by mělo zamezit obrovskému množství pohybu diskových hlav, jak je tomu u běžného sekvenčního kopírování. To znamená, že kopírování bude rychlejší. Jestliže se kopírují data mezi dvěma úložišti (například mezi dvěma disky), program kopíruje pomoci několika vláken, která zároveň čtou i zapisují.



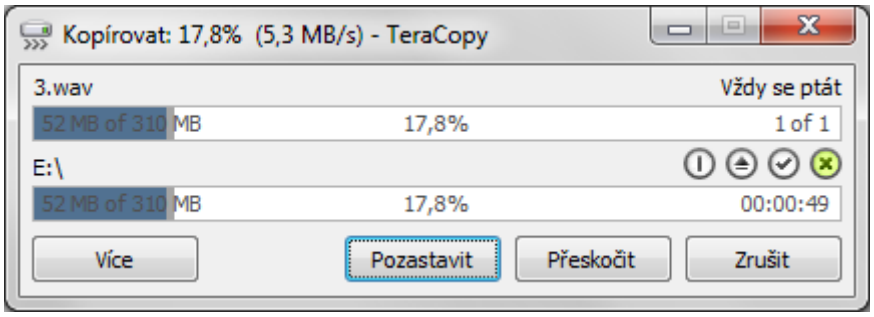
Obrázek 3.2: Grafické rozhraní programu FastCopy

Uživatelské rozhraní nepůsobí příliš přívětivě, ale jeho funkce to bohatě vyváží. Navíc je toto rozhraní docela jednoduché a po čase si člověk zvykne.

Program nabízí možnost volby způsobu kopírování, buď s použitím vyrovnávací paměti, nebo s použitím vláken nebo automatické volby. Dá se nastavit, i co se má s kopírovanými soubory provést v případě, že již soubor existuje. Zda-li se má soubor vždy přepsat nebo nekopírovat, jestli se mají přepsat ty, které se liší datem či velikostí. Na výběr je i možnost soubory pouze přesunout, nebo smazat. Další nastavení je například možnost Nonstop, která v případě chyby během kopírování více souborů nezastaví kopírování, ale kopíruje ostatní soubory až do konce, a až po ukončení zpracovávání všech programů vypíše chybovou hlášku. Program má také možnost synchronizace či vytváření jobů nebo integraci do kontextového menu průzkumníku. V archivu s programem je i přibalený soubor pro instalaci, ale tento program ani není nutno instalovat, lze jej rovnou spustit.

3.3 TeraCopy

TeraCopy je freeware program, který slouží podobně jako FastCopy a Robocopy, ke kopírování souborů. TeraCopy slouží k rychlejšímu kopírování a přemisťování souborů na disku, popř. mezi disky. Tento program je navržen tak, aby tyto operace probíhaly za co největší možné rychlosti. TeraCopy nekopíruje chybné soubory, ale přeskočí je a na konci procesu kopírování zobrazí uživateli dané soubory, které byly přeskočeny. TeraCopy umožňuje automaticky kontrolovat kopírované soubory a hledat chyby pomocí CRC(cyclic redundancy check). Existuje také možnost si nastavit TeraCopy jako výchozí program pro kopírování a přesouvání souborů. Tento program je podporován i ve Windows 8.



Obrázek 3.3: Grafické rozhraní programu TeraCopy

Existuje i placená verze tohoto programu, která obsahuje rozšíření, které zahrnuje ukládání a přemisťování do oblíbených složek, ukládání zpráv do formátu HTML, či odstranění vybraného souboru z fronty na kopírování.

Pro rychlejší kopírování využívá TeraCopy dynamicky proměnný buffer. Asynchronní kopírování také zrychluje kopírování mezi dvěma pevnými disky. TeraCopy také obsahuje možnost k pozastavení kopírování, které můžeme využít k uvolnění disku. Když se rozhodneme pokračovat v kopírování, pokračujeme v místě, kde jsme skončili. Je to výborná pomůcka, když potřebujeme uvolnit zdroje disku, ale nechceme začínat s kopírováním od začátku.

Při chybě se TeraCopy pokouší opakovaně kopírovat daný soubor. Pokud tyto pokusy opakovaně selžou, TeraCopy přeskočí daný soubor a dokončí přenos ostatních, nepoškozených souborů. Po skončení se vypíší chybové hlášky a zobrazí se nám soubory, které se nepovedlo zkopírovat.

4 Historie jazyka C#

Praktická část této práce je implementována v jazyce C#. Dále si tento jazyk blíže popíšeme. Tento programovací jazyk se vyvinul na základě jazyků C a C++. Jazyk C je velmi používaným programovacím jazykem, jehož využití je veliké. Jazyk C vznikl pro potřeby vyvinutí operačního systému UNIX. Jeho jádro je napsáno výhradně v jazyce C. V současné době se využívá nejvíce pro psaní systémového softwaru, ale využití můžeme najít i u aplikací. Nespornou výhodou je importování jazyka symbolických adres (Assembly language) do kódu. Jazyk C je přenositelný i na jiné architektury. Mnoho jazyků, mezi které patří i Java či PHP, si převzaly syntaxi z jazyka C, neboť je lehce pochopitelná a převážná část operačních systémů, knihoven či překladačů a interpretů je implementována pomocí jazyka C. Později přišlo rozšíření v podobě jazyka C++, které je již objektově založené na rozdíl od jazyku C. Podporuje také procedurální programování i generické. Jazyk C je podmnožinou jazyka C++, což znamená, že programy napsané v jazyce C vesměs fungují i v jazyce C++, ale bohužel to neplatí vždy, existují i výjimky, které překladače pro jazyk C++ nepřeloží. Zpočátku byl jazyk C++ překládán do jazyka C. V současné době je jazyk C++ jedním z nejrozšířenějších.

V tomto odstavci si povíme něco o novějším jazyku C#. Tento programovací jazyk byl založen na jazycích Java a C++, což znamená, že je nepřímým potomkem jazyka C. Tento jazyk čerpá syntaxi z jazyka C. C# je vysokoúrovňový objektově orientovaný programovací jazyk, který byl vyvíjen společně firmou Microsoft, která vyvíjí operační systémy Windows a platformou .NET Framework. Na rozdíl od jazyka C má několik výhod, mezi které patří podpora pro hlídání hranice polí, detekce použitých neinicializovaných proměnných a automatický garbage collector, který zajišťuje využití již nepoužívané části paměti vyhrazené programem.

Vývoj tohoto jazyka začal již v roce 2002, kdy se implementovala pouze základní podpora objektů na základě zkušeností nasbíraných v jazycích Java a C++. Další verze C# vyšla až v roce 2005, kdy se tento jazyk dočkal mnoha změn. Mezi nejdůležitější patří podpora generich, částečných a statických tříd, iterátorů, anonymní metody pro lepší používání delegátu neboli odkazů na metody, ale také podpora operátoru koalescence a nullovatelné hodnotové typy. Další verze vyšla v roce 2007, kdy byla také vydána nová podpora platformy .NET Framework 3.5 a softwarová aplikace pro vývoj Visual Studio 2008. Mezi změnami v tomto novém vydání jazyka C# najdeme lambda výrazy, které jsou také v dalších jazycích jako je Python, aj. Díky nim můžeme vytvářet anonymní metody, které obsahují jeden výraz, nebo několik příkazů, to pak můžeme použít v chvílích, kdy bychom mohli očekávat instanci delegáta. Kvůli lambda výrazu musel být do jazyka implementován nový operátor „=>“, který nazýváme „přechází v“. Nastalo také zjednodušení při zapisování proměnných, či rozšiřujících metod. V nejnovějším vydání tohoto programovacího jazyka se objevila kovariance a kontra variance, dynamické datové objekty, volitelné parametry a pojmenované parametry.

C# je podporován v několika vývojových prostředích, mezi které například patří: Visual Studio, Turbo C# Explorer, SharpDevelop, MonoDevelop a pro NetBeans a Eclipse lze doinstalovat plug-in s podporou C#.

Jazyk je vhodný pro vývoj aplikací, databázových programů, webových aplikací a služeb, softwaru pro mobilní zařízení a mnoho dalších.

Jazyk C# lze využít i pro vývoj operačního systému. Například operační systém Singularity je z části napsán v jazyce C#. Dokonce existuje kompilátor jazyka C# za pomoci kterého je možné v C# naprogramovat operační systém. Tento projekt se jmenuje cosmos.

5 Program

Praktická část této práce byla vyvíjena v jazyce C# a platformě .NET Framework. Programovací jazyk C# jsem si vybral, protože s ním mám nejvíce zkušeností. Na výběr připadaly i další jazyky jako Java či C++, ale já zvolil C#. Program je vytvořen pro .NET Framework verze 4, tudíž program půjde spustit všude, kde bude framework této verze nainstalována. Tuto verzi frameworku lze nainstalovat na operační systémy Windows XP SP3, Windows Server 2003 SP2, Windows Vista SP1 nebo novější, Windows Server 2008, Windows 7.

5.1 Cíl programu

Úkolem je vypracovat aplikaci pro systém Windows, která umožňuje kopírovat soubory těmito variantami:

1. Sériové čtení se sériovým zápisem.
2. Sériové čtení s paralelním zápisem.
3. Paralelní čtení se sériovým zápisem.
4. Paralelní čtení s paralelním zápisem.

Program by měl k tomuto využívat vyrovnávací paměť. Velikost vyrovnávací paměti se určí na základě testování s rozdílnými hodnotami velikosti této vyrovnávací paměti. Velikost je pevně určena programem. Také se soubory budou zpracovávat ve vláknech, aby bylo možné pracovat se soubory paralelně.

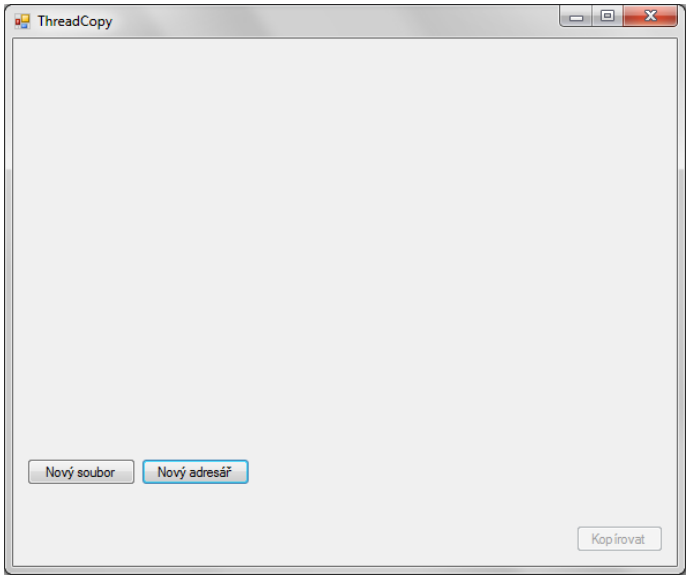
5.2 Funkce programu

V této kapitole se budeme zabývat funkcí daného programu. Popíšeme si, jak se s programem pracuje a jakým způsobem program funguje.

5.2.1 Grafické rozhraní

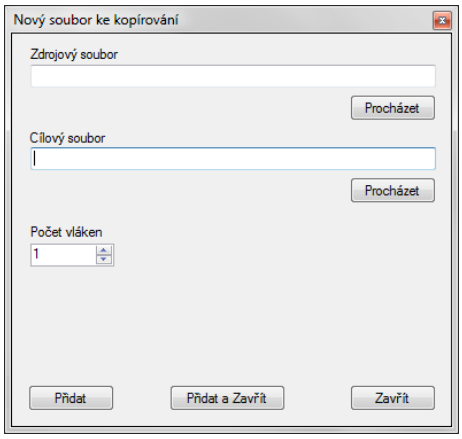
Aplikace je určena pro systém Windows, z tohoto důvodu aplikace obsahuje jednoduché grafické rozhraní pro snažší ovládání místo konzolového rozhraní. Toto grafické rozhraní neobsahuje vlastní metody pro kopírovaní souboru. Veškerá logika je uložena v dynamické knihovně, kterou obsluhuje samo grafické rozhraní. Díky tomu lze vytvořit jiné libovolné rozhraní pro ovládání aplikace. Nyní se podíváme na samotné rozhraní, které jsem navrhl a které je součástí praktické části bakalářské práce.

Rozhraní je jednoduché a využívá všechny možnosti aplikace. Hlavní okno aplikace obsahuje tlačítka pro přidáváni a odebíraní souborů určených ke kopírování, komponentu list box se seznamem souborů, check box pro možnost zapisování výsledků do souboru, popisky pro popis funkcí komponent, zobrazení nastavení a výpisu aktuálně prováděné akce či výpisu výsledku. Dále tlačítko pro spuštění kopírování a progress bar, pro zobrazeni průběhu kopírováni.



Obrázek 5.1: Okno aplikace po spuštění

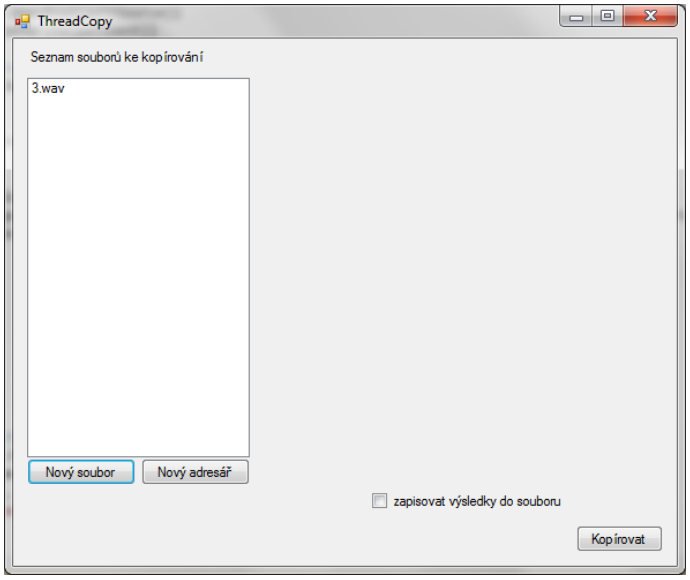
Rozhraní pracuje tak, že zobrazuje jen komponenty, které můžeme aktuálně použít. To znamená, že po spuštění je formulář téměř prázdný, protože jsou skoro všechny prvky skryty, až na tlačítka pro přidání souboru a tlačítka pro zahájení kopírování, které je zašedlé a nejde v tuto chvíli zmáčknout. Pro přidání souborů slouží dva tlačítka, jedno pro přidání jednoho souboru, druhé pro přidání celého adresáře.



Obrázek 5.2: Okno pro přidání souboru ke kopírování

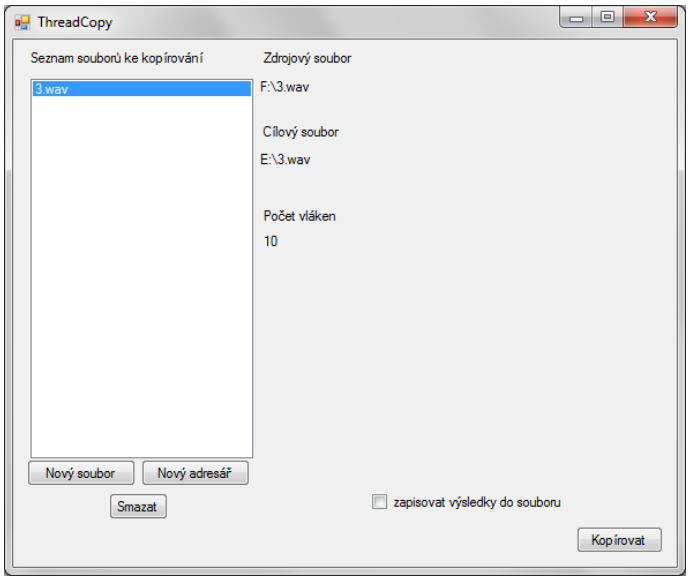
Po kliknutí na tlačítko „Nový soubor“ se nám otevře dialogové okno s nastavením pro přidání záznamu do seznamu kopírovaných souborů. Toto okno obsahuje pole „Zdrojový soubor“ s názvem zdrojového souboru, tedy souboru, který se má zkopírovat, pole s názvem sílového souboru, pole pro zadání množství vláken, které má soubor zpracovávat a tlačítka pro potvrzení nebo zavření okna. Cestu k souboru, který chceme kopírovat, buď ručně vypíšeme do políčka, nebo klikneme na tlačítko „Procházet“ a otevře dialogové okno pro výběr souboru, kde soubor vybereme. V poli „Cílový soubor“ je cesta, kam se má soubor zkopírovat, vybírá se stejným způsobem jako zdrojový soubor. Dále si můžeme vybrat počet vláken, které mají zpracovávat tento soubor. Nakonec můžeme volbu potvrdit tlačítkem „Přidat“, které přidá záznam do seznamu kopírovaných souborů a nechá okno otevřené, pro možnost přidání dalšího souboru. Jestliže zmáčkneme tlačítko „Přidat a Zavřít“, do

seznamu souborů určených ke zkopírování se přidá záznam a okno se zavře. Pro zrušení volby souboru stiskneme tlačítko „Zavřít“, které pouze uzavře toto okno a neuloží zadané informace.



Obrázek 5.3: Okno aplikace se záznamem

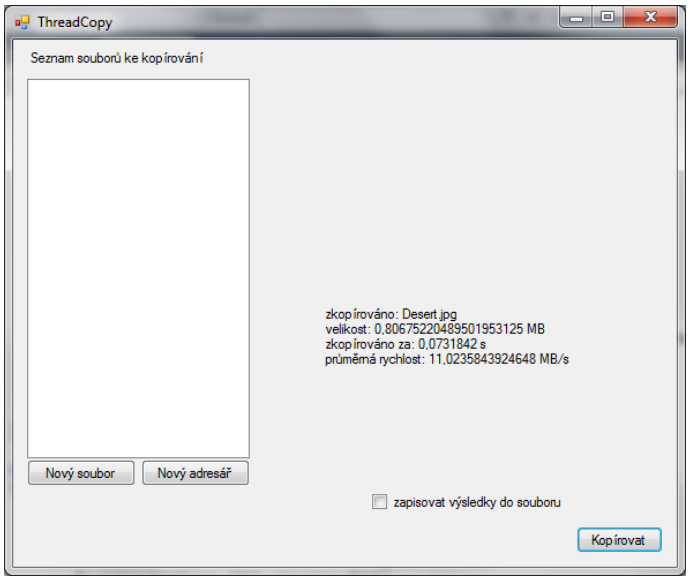
Po kliknutí na tlačítko „Nový adresář“ se otevře podobné okno, jako v předchozím případě. Jediný rozdíl je ve vybírání souborů, v tomto případě adresářů. Do políčka „Zdrojový adresář“ se zadá cesta ke složce, jejíž obsah se má zkopírovat. V případě, že nechceme zadávat cestu ručně, můžeme zvolit složku pomocí tlačítka „Procházet“, které zobrazí dialogové okno pro výběr adresářů. Podobně je tomu u políčka „Cílový adresář“. Dále je postup stejný, jako v případě kopírování jednoho souboru. Po přidání adresáře ke kopírování se přidá každý soubor ze zadaného adresáře zvlášť. Soubory uložené v podsložkách se zkopírují včetně zanoření.



Obrázek 5.4: Okno aplikace s označeným záznamem

Po přidání souboru ke kopírování se v hlavním okně aplikace zobrazí list box se seznamem souborů, které se budou kopírovat. Navíc také přibude tlačítko „Smazat“ pro možnost smazání

vybraného souboru ze seznamu kopírovaných souborů. Máme také možnost kliknutím vybrat libovolný soubor v list boxu a vpravo se zobrazí nastavení kopírování tohoto souboru. Objeví se zaškrtávací tlačítko „zapisovat výsledky do souboru“, které po kopírování vytvoří soubor log.txt, nebo jestliže již existuje, tak do něj připíše záznam o proběhlém kopírování. V tomto záznamu je uveden čas, kdy kopírování proběhlo, jméno kopírovaného souboru, rychlost kopírování, velikost kopírovaného souboru a doba za kterou byl soubor zkopírován. Nakonec se také zapne tlačítko „Kopírovat“, které po kliknutí začne kopírovat veškeré soubory, které jsou uvedeny v seznamu souborů ke kopírování. Pokud se zkopírují veškeré soubory, nebo se odstraní za pomocí tlačítka smazat, zmizí tlačítko „Smazat“, zašedne tlačítko „Kopírovat“ a zmizí zaškrtávací tlačítko „zapisovat výsledky do souboru“.



Obrázek 5.5: Zobrazení výsledků

Po kliknutí na tlačítko „Kopírovat“ se začnou kopírovat soubory, které jsou v seznamu. Soubor se vymaže ze seznamu kopírovaných souborů. Zobrazí se zpráva, že se kopíruje daný soubor a zobrazí se progress bar, který ukazuje stav kopírování. Po zkopírování souboru se zobrazí zpráva o průběhu. Jestli je v seznamu další soubor, okamžitě se začne kopírovat, dokud se nezkopírují všechny soubory.

5.2.2 Logika programu

Program je rozdělen na oddělenou logiku a grafické rozhraní. Takovéto rozdělení umožňuje vytvořit nová rozhraní k této logice. Při začátku implementace již bylo přibližně jasné, jak bude grafické rozhraní vypadat. Bylo zapotřebí ukládat parametry kopírování, a to je jméno souboru i s cestou, cesta k adresáři a jméno nového souboru a počet vláken, které budou soubor zpracovávat. Navíc zobrazovat seznam těchto souborů v komponentě list box. Z tohoto důvodu byl pro ukládání těchto dat využit právě list box. Itemy list boxu jsou uloženy v kolekci ObjectCollection, to znamená, že se do kolekce dá vložit libovolný objekt. Pro uložení parametrů bylo tedy třeba vytvořit třídu, která by měla tyto parametry. Tato třída obsahuje konstruktor s parametry kopírování, které mají metodu get s modifikátorem public. Ještě je třeba, aby se v list boxu vypisoval srozumitelný text, aby uživatel věděl, co se to vůbec zobrazuje. Jako rozumné řešení je zobrazovat jméno souboru, které se má

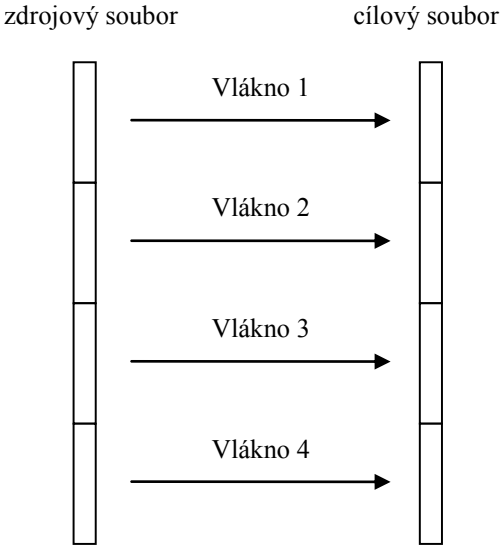
kopírovat. V list boxu se zobrazuje hodnota z metody ToString() jednotlivých itemů. Z tohoto důvodu má třída metodu ToString(), která z celého jména souboru vrací pouze jméno souboru.

Komponenta list box se má chovat tak, že po označení nějakého itemu se zobrazí v několika komponentách typu label parametry kopírování. Z tohoto důvodu je list box nastaven, aby bylo možné označit pouze jeden item. Toto je vlastnost SelectionMode nastavena na možnost One. Pro vypsání parametrů do labelů je třeba využít událost list boxu SelectedIndexChanged, která se vyvolá při změně označení itemu v list boxu. V metodě přiřazené k této události je pouze podmínka, zda-li je vůbec označen libovolný item. Jestliže je označen, z vybraného itemu se uloží parametry do příslušných labelů a ty se zviditelní, v případě že není označený žádný item, například při smazání itemu, nebo v případě prázdného list boxu, se tyto labely skryjí.

Mazání itemů z list boxu je možné pomocí tlačítka „Smazat“. Tlačítko se zobrazí, pouze pokud je vybrán item, jinak je skryté. Jeho funkce je jednoduchá, pouze odstraní vybraný záznam z list boxu, a v případě že už žádný item v list boxu není, skryje list box a jeho popisek, check box pro ukládaní výsledků do souboru au tlačítka „Kopírovat“ nastaví Enabled na false.

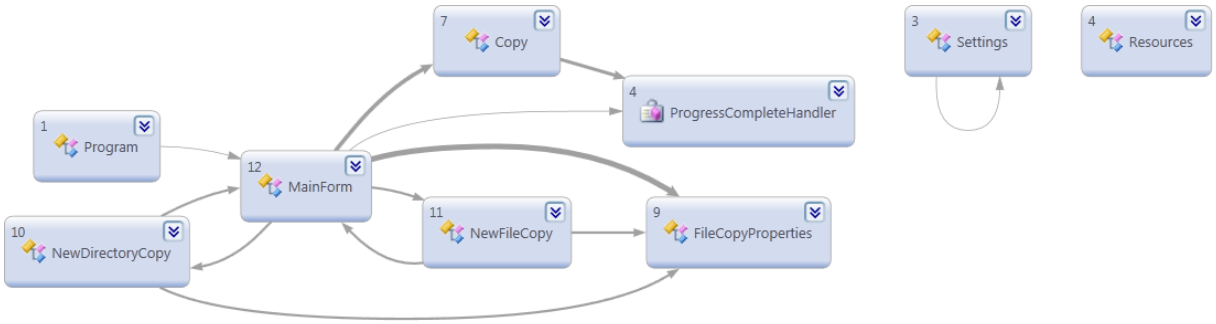
Zahájení kopírování se provádí tlačítkem „Kopírovat“. Po stisku tohoto tlačítka se zjišťuje, jestli není seznam prázdný. Jestliže seznam obsahuje nějaké itemy, vytvoří se instance třídy z logické části a nastaví se událost CopyComplete, která se vyvolá po ukončení kopírování. Tato událost spustí metodu, která vypíše výsledek kopírování a zavolá opět metodu kopírování. Poté se zavolá metoda pro kopírování souboru s parametry prvního itemu v list boxu a předává se i metoda pro zobrazování průběhu kopírování. Nakonec se smaže první item v list boxu a zviditelní progress bar pro zobrazování výsledků.

Dále se podíváme na samotnou třídu logiky a podíváme se, jak pracuje. Načtou se informace o souboru pomocí třídy FileInfo. Zjišťuje se, jestli opravdu soubor existuje, pokud ne, tak se vyvolá chyba. Hlavně se ale zjišťuje velikost souboru. Jestliže je soubor moc malý a vyrovnávací paměti všech vláken jsou větší než samotný soubor, změní se velikost vyrovnávací paměti. Poté se spustí měření času pomocí třídy Stopwatch a v cyklu spouští vlákna, která asynchronně kopírují část souboru. Jako parametr se jim předá pozice, od které se má kopírovat a délka kopírovaného bloku. Pro práci s vlákny jsem použil třídu BackgroundWorker, která tuto práci usnadňuje a nabízí více možností. Po ukončení zpracovávání se zavolá metoda Complete, která ukončí měření času a vytvoří zprávu o výsledku. Jestliže je zapnutá volba ukládání výsledku do souboru, tak se tato zpráva přidá do souboru log.txt v adresáři aplikace. Dále se vyvolá událost CopyComplete a jako parametr se předá zpráva o výsledku. Grafické rozhraní reaguje na tuto událost spuštěním přiřazené metody. V této metodě se pouze do popisku vypíše zpráva o výsledku kopírování a spustí se znovu metoda kopírování. V této metodě se opět volá nové kopírování, jestliže existují další soubory v seznamu nebo se metoda ukončí.



Obrázek 5.6: Kopírování souboru pomocí vláken

Kopírování programu pomocí vláken je znázorněno na obrázku 5.7. Zdrojový soubor se rozdělí na stejné díly, podle nastavení v aplikaci. Vytvoří se cílový soubor, do kterého se bude kopírovat zdrojový soubor. Každé vlákno kopíruje jen určitou část souboru nezávisle na ostatních. Kopírování v rámci jednoho vlákna probíhá sekvenčně. Znamená to, že se načte část zdrojového souboru do vyrovnávací paměti a poté se zapíše z vyrovnávací paměti do cílového souboru.



Obrázek 5.7: Třídní diagram aplikace

5.3 Chyby

Zde si popíšeme chyby a nedostatky tohoto programu a technické zdůvodnění, proč jsou tyto nedostatky součásti této aplikace.

Hlavním nedostatkem této aplikace je skutečnost, že nedokáže číst sériově a zapisovat paralelně, nebo naopak, číst paralelně a zapisovat sériově. Tyto způsoby jsou vyžadovány v zadání práce, ale v programu nejsou implementovány. Je to z toho důvodu, že paralelní zápis do souboru je Thread Safe. To znamená, že při zapisování vláken běží jedno po druhém. Z toho vyplívá, že se prakticky jedná o sériový zápis za všech okolností. To znamená, že změnou počtu pracovních vláken se mění paralelizace čtení, ne zápisu. Z tohoto důvodu aplikace neumožňuje nastavení sériové čtení paralelní zápis a naopak.

5.4 Testování a výsledky

V této kapitole se budeme zabývat testovaním aplikací. Pro testování je třeba zvolit testy, které by ukázaly výkon jednotlivých aplikací. Je zapotřebí testovat i jiná média, než je obyčejný pevný disk. Mezi dále testované média patří optická mechanika (CD/DVD) a elektronická paměť v podobě flash disku. To znamená, že nám nabízí devět kombinací kopírování. Vzhledem k tomu, že nemůžeme zapisovat na optický disk přímo, sníží se počet možných kombinací testování o zápisy z všech tří druhů pamětí.

Pro testování rychlosti jsem vybral jeden větší soubor uložený na optickém disku. Tento soubor zkopírujeme na pevný disk i na flash disk, dále vytvoříme kopii tohoto souboru na stejném disku. V poslední fázi testování zkopírujeme soubory z pevného disku na flash a naopak.

Pro měření kopírování je třeba měřit dobu, za kterou se soubor zkopíruje. Některé programy měří dobu kopírování. Robocopy zobrazuje čas, kdy kopírování začalo a kdy skončilo. Program také vypisuje čas potřebný ke kopírování, ale ten je rozdělen na více hodnot, tudíž je nejjednodušší možnost odečítat čas začátku a čas konce kopírování. Programy FastCopy a Teracopy přímo zobrazují dobu, za kterou se program zkopíroval. Aplikace vytvořena pro praktickou část této práce také zobrazuje čas potřebný ke zkopírování souboru. Jediný problém je, jak měřit čas kopírování Průzkumníkem Windows. Řešením tohoto problémů je jedině sledovat systémový čas, nebo použít stopky.

Během zkoušení programu bylo zjištěno, že větší výkon se dosáhne s vyšším počtem vláken. Velikost vyrovnávací paměti byla nastavena v programu FastCopy i v aplikaci k této práci na 32MB. Počet vláken během testování nastaven v aplikaci vytvořené pro tuto práci na 10. V příkazu robocopy byl použit parametr /mt, který umožňuje kopírovat ve vláknech. Nastavení vláken bylo necháno v původním nastavení, to je 8 vláken. Všemi programy byl kopírován stejný soubor o velikosti 728 MB.

Tabulka 1: Výsledky kopírování

	Průzkumník	TeraCopy	FastCopy	Robocopy	aplikace k této bakalářské práci
DVD>Flash	2:31	2:29	2:27	2:24	**2:21**
DVD>HDD	2:50	2:47	**2.17**	2:54	2:25
Flash>Flash	3:09	**3:00**	3:03	3:03	3:05
HDD>HDD	2:57	2:33	2:38	**2:29**	2:37
Flash>HDD	2:50	2:38	2:52	2:23	**2:19**
HDD>FLASH	2:25	**2:23**	2:26	2:24	2:28
HDD>2.HDD	1:56	1:44	1:36	1:33	**1:27**

Během testování bylo zjištěno, že okno kopírování Průzkumník Windows zobrazuje matoucí informace o kopírování. Zobrazovaný čas zbývající do konce je chybný, stejně jako ukazatel průběhu. Práce s Robocopy není snadná pro běžného uživatele, protože nemá grafické rozhraní, ale pro nastavení automatického zálohování je ideální.

5.5 Možnost rozšíření programu

Tento program není původně určen pro každodenní práci. Cílem tohoto programu je zjistit, jestli je možné vůbec kopírování souborů zrychlit, a jaký má vliv nastavení vyrovnávací paměti a počet zpracovávajících vláken na rychlost kopírování. Z tohoto důvodu má aplikace dost co zlepšovat pro běžné užívání. Samotné přidávání souborů je docela zdlouhavé a neohrabané. Zjednodušení by mohlo být například možnost přidávání pomocí přetáhnutí souboru do aplikace pomocí kurzoru myši. V nejlepším případě by program mohl nahradit původní utilitu operačního systému Windows pro kopírování.

Další možnost rozšíření by mohlo být automatická detekce typy disků, mezi kterými se bude kopírovat, nebo jestli se bude kopírovat na stejný disk. Podle toho by aplikace mohla změnit způsob nebo nastavení kopírování, pro větší výkon.

6 Závěr

Cílem této práce bylo vytvořit aplikaci, která by dokázala zrychlit kopírování souborů, oproti Průzkumníku Windows, za použití vláken a vyrovnávací paměti. Dále aplikaci porovnat s podobnými programy zabývající se stejnou problematikou.

Na začátku práce jsme se seznámili s problematikou zrychlení kopírování souborů z teoretické stránky, poté i z praktické. V popisu problematiky jsme se zmínili o vláknech, které bude program využívat pro kopírování souborů. Poté jsme si tato vlákna popsali, a spolu s nimi jsme se zmínily i o multitaskingu, díky kterému lze vlákna využívat.

V bakalářské práci jsou taky zmíněny programy, které se zabývají stejnou problematikou a jsou blíže popsány. Tyto aplikace byly testovány a porovnávány s praktickou částí této bakalářské práce.

Nejdůležitější částí bakalářské práce bylo testování vytvořené aplikace. Součástí bylo porovnání s ostatními programy představených v úvodu práce, které byly taky testovány. Z výsledků testů, podle očekávání, vyplynulo, že Průzkumník Windows kopíruje soubory nejpomaleji. Porovnání ostatních programů už není tak jednoznačné, ale ve většině případů je naše aplikace lepší, i než konkurenční programy. Z těchto výsledků vyplívá, že má smysl zabývat se kopírováním souborů na PC. Uživatelsky i výkonově je nejideálnější program TeraCopy, protože nahrazuje kopírovací nastroj Průzkumníku Windows. Aplikace vytvořena k této práci by se mohla uživatelsky upravit, byla by tak schopna konkurovat podobným aplikacím. Možná i optimalizacemi a vyladěním ještě urychlit. Funkce této aplikace může být výrazně ovlivněna novými technologiemi v oblasti ukládání dat, které mohou znamenat, že způsob zpracovávání bude neefektivní. Mezi novější technologie patří SSD disky, které nejsou novinkou, ale cenově už jsou dostupné a začínají se využívat čím dál tím častěji. V případě disků SSD by měla mít aplikace dobré výsledky. Otázkou tedy je, kdy se objeví na místo SSD nový druh úložiště a jakým způsobem bude pracovat.

Použitá literatura

[1] DEMBOWSKI, Klaus; *Mistrovství v HARDWARE.* 1.vydání, v Computer press, 2009. 712 s. ISBN 80-251-0416-8

[2] HANÁK, Ján; *C# 3.0: programování na platformě .NET 3.5.*1. vydání, v Zoner Press, 2009. 282 s. ISBN 8074130460

[3] ROBINSON, Simon; *C#: programujeme profesionálně*.1. vydání v Computer press, 2003. 1130 s. ISBN 8025100855

[4] RACEK, Stanislav; JEŽEK, Karel. *Paralelní programování*, 1.vydání, Západočeská univerzita, 1994. 176 s. ISBN 8070821280

[5] Magazín Stahuj.cz [online]. 2009 [cit. 2009-10-1]. Magazín Stahuj.cz. Dostupné z WWW: http://magazin.stahuj.centrum.cz/ssd-disky-budoucnost/

[6] http://msdn.microsoft.com/library/

[7] http://en.wikipedia.org/wiki/Ssd

[8] http://en.wikipedia.org/wiki/USB_flash_drive

[9] http://poli.cs.vsb.cz/edu/apps/down/disky.pdf

Přílohy

Seznam příloh

Příloha A: Adresářová struktura přiloženého DVD

/bakalarska-prace-priloha-chl159.zip	Zdrojové kódy aplikace včetně zkompilované programu, která je zkomprimována